TIE-02500 Rinnakkaisuus, projekti 3

Petri Leppänen, 160280

Jorma Syrjä, 273006

THERAC-25

Artikkelin mukaan järjestelmässä oli kaksi pääongelmaa rinnakkaisuuden kannalta.

Ongelma 1: Pääohjelman kanssa samanaikaisesti ajetaan tehtävää Datent (Data entry). Ohjelmassa on lippu (jaettua dataa), joka kertoo pääohjelmalle, että tietojen syöttö on valmis. Kun huomataan, että lippu on aktiivinen, siirrytään pääohjelmassa hoidon seuraavaan vaiheeseen, joka on pitkäkestoinen (esimerkkitapauksessa 8 sekuntia).

Jos tämän tehtävän suorituksen aikana muutetaan (Datent-aliohjelmassa) parametreja, niin toinen lippu, joka kertoi parametrien muuttuneen, ei tullut tarkastetuksi. Tämä ongelma aiheutti sen, että esimerkkitapauksessa ohjelma jatkoi väärään hoitomuotoon. Samaan aikaan operoijan päätteellä (tässä vaiheessa huoneen ulkopuolella) saattoi näkyä korjatut parametrit, vaikka ne eivät oikeasti olleet voimassa pääohjelmassa. Tämä johtui siitä, että operoija oli nuolinäppäimillä mennyt muuttamaan syötettyjään arvoja ja nopeasti kuitannut kaikki arvot enterillä uudestaan järjestelmään.

Ongelma 2: Ylivuoto jaetussa muuttujassa mahdollistaa sen, että tärkeä tarkastus ohitetaan ja mahdollinen virhetilanne jää huomaamatta. Kun operoija on syöttänyt hoidon parametrit ja varmistanut ne hoitohuoneen päätteellä, järjestelmä antaa viestin "PRESS SET BUTTON", joka ohjeistaa operoijaa painamaan SET nappia laitteen kyljessä olevasta ohjaimesta, tai kirjoittamaan päätteeseen SET.

Samaan aikaan kun ohjelma odottaa napin tai komennon syöttöä käyttäjältä, taustalla pyörii toinen aliohjelma, Set Up Test, jota kutsutaan uudelleen niin monta kertaa kuin ehditään, ennen kuin käyttäjä painaa SET-nappia tai kirjoittaa päätteelle set. Ongelma syntyy Set Up Test-aliohjelmassa, jossa joka kierroksella kasvatetaan jaettua muuttujaa. On mahdollista, että käyttäjä painaa SET-nappia juuri silloin kun ollaan menossa 256:tta kierrosta, jolloin 8-bittinen muuttuja ylivuotaa ja pyörähtää nollaksi.

Tämä on ongelma, koska toisessa samanaikaisesti ajavassa aliohjelmassa, joka tarkistaa jaetun muuttujan arvoa indeksinä käyttäen jotain turvallisuuteen liittyvää asetusta toisaalla. Tätä tarkistusta ei tehdä, jos muuttujan arvo on 0. Täten on mahdollista, että jokin tärkeä tarkistus jätetään tekemättä, jos käyttäjä painaa set-nappia "väärään" aikaan.

Rinnakkaisuuden kannalta aihetta miettiessä koska järjestelmässä on useita kriittisiä ja ei-kriittisiä tehtäviä, niin irrottava vuoronnus itsessään voi aiheuttaa rinnakkaisuusongelmia eli tärkeä tehtävä voidaan keskeyttää juuri sillä hetkellä, kun tärkeää jaetun datan muuttujan arvoa ollaan säätämässä (näitä on tässä järjestelmässä paljon). Artikkelissa myös mainitaan suoraan, että ohjelmistossa on useita jaettuja muuttujia, joiden kirjoitusta tai lukua ei synkronoida millään tavalla.

Lisäksi järjestelmässä oli muita suuri puutteita ja ongelmia. Esimerkiksi ohjelmallisesti ja käyttöjärjestelmän kannalta suurin ongelma oli se, että kursorin sijainti operoijan päätteellä vaikutti siihen, että meneekö päivitetyt tiedot operoijan päätteeltä pääohjelmalle oikein.

Kokonaisuudessaan ohjelmistontuotannon kannalta laitteen ohjelmistossa oli tehty suuria virheitä. Mekaanisia varmistuksia ei tehty enää 25-versioon, vaan luotettiin täysin koneen operoivaan ohjelmaan. Myös joitain ohjelman osia kierrätettiin vanhoista versioista, tietoisesti mutta myös muita tietämättä jotka yhdessä vaikutti siihen että ohjelmasta tuli niin buginen.

Myös manuaaleissa oli suuria puutoksia. Esim. virhekoodien selitykset puuttuivat manuaalista joista osa virhekoodeista olisi kerrottu suoraan, että potilaat olivat vaarassa. Artikkelissa myös suoraan sanottiin, että laitteen ohjelmistosta oli mahdotonta löytää kaikkia bugeja. Myös se oli käsittämätöntä, että turvallisuusanalyysi oli tehty, mutta vain mekaniikan osalta. Lisäksi ensimmäisten onnettomuuksien ja järjestelmän korjauksien jälkeen järjestelmää oli korjattu, mutta testejä ei dokumentoitu lainkaan.

Ongelmia toi myös se, että fyysisesti Therac-25:ssa on kaksi päätettä, yksi hoitosalin sisällä ja toinen ulkopuolella. Hoidon suorittamisen aikana operoija joutuu käyttämään molempia päätteitä ja vain osaa arvoja voi syöttää salin puolella. Loput arvot syötetään valvontahuoneessa olevalla päätteellä, jossa on myös lukujen varmistus ja jossa muutenkin operoidaan laitetta.

Artikkelissa annetun analyysin perusteella suuri osa ohjelmistossa esiintyvistä ongelmista liittyy jaetun datan väärinkäyttöön. Kurssilla olemme käyneet läpi useita tapoja käsitellä jaettua dataa: mutex, semafori, jonot jne. Ohjelmisto on mahdollista korjata näitä tietoja hyväksikäyttäen.

Toinen asia on sitten se, onko pelkillä koodikorjauksilla mahdollista tehdä laitteesta turvallinen. Kuten artikkelissakin mainittiin, Therac-25 luottaa täysin ohjelmistoon, jonka vastuulla on kaikkien varmistimien tarkistus. Aikaisemmissa Therac-laitteissa varmistimet tehtiin yhteistyössä ohjelmiston ja raudan kanssa ja pääasiassa luotettiin fyysisiin mekaanisiin lukkoihin.

Kyseessä on kuitenkin laite, joka pommittaa potilasta suurilla määrillä säteilyä. Laitteen oikean toiminnallisuuden varmistamiseen tarvitaan sellaista tietoa, jota ohjelmoijalta ei välttämättä löydy. Nämä seikat huomioon ottaen, en hoidattaisi itseäni laitteella pelkkien ohjelmistokorjausten jälkeen. Tämä operaatio oli todella surullinen esimerkki siitä, miten ohjelmistoja ei pitäisi tehdä. Tai oikeastaan tämä oli todella hyvä esimerkki siltä osin katsoessa.

Ohjelmiston korjaamiseen kyllä voisi löytää korjaustoimenpiteitä. Ensimmäinen vaihtoehto ohjelmiston korjaamiselle olisi muokata olemassa olevaa koodia seuraavanlaisesti:

- Kaikki jaetun datan käsittely synkronoitaisiin lukituksilla. Tietorakenteiden monimutkaisuudesta ei ole tässä tietoa, mutta oletetaan, että poissulkeminen riittää.
- Kaikki silmukat, joissa odotetaan jonkin lipun päivittymistä, kirjoitetaan uudelleen niin, että käytetään wait-notify –rakenteita.
- Pitkäkestoisten tehtävien aliohjelmat tulee kirjoittaa uudelleen niin, että aliohjelmassa suoritetaan do...while -tyyppistä silmukkaa, jossa joka kierroksella tarkistetaan jonkinlainen pysäytysehto (Cancellation token)

Toinen ja aika varmasti oikeampi tapa korjata ohjelmisto olisi kirjoittaa se uudelleen jollakin ohjelmointikielellä, joka noudattaa Communicating Sequential Processes –suunnittelua. Tällä saataisiin kaikki jaetun datan aiheuttamat ongelmat poistettua. Artikkelin mukaan vakavimmat ongelmat Therac-25:ssa aiheutuivat nimenomaan jaetun datan synkronoimattomuudesta, joten tämä olisi paras vaihtoehto.

Suojaus ohjelmistovirheiltä pitäisi olla sisäänrakennettu järjestelmään ja itse ohjelmistoon. Koska aina emme voi poistaa kaikkia ohjelmistovirheitä, niin ohjelmiston lisäksi pitää järjestelmään lisätä paljon myös muita varmistuksia vaikkakin ne olisi ohjelmallisesti jo hoidettuna. Varmistamisella voidaan näinkin tärkeissä asioissa estää ongelmien pahimmat vaikutukset ja estää niiden syntyminen kokonaan. Lisäksi pitää ohjelmistosuunnitteluun panostaa hurjasti enemmän mitä tässä tapauksessa on alun perin tehty. Ohjelmisto pitää suunnitella tarpeeksi yksinkertaisesti, jotta se on ymmärrettävissä ja että se olisi myös testattavissa.